CPE301 – SPRING 2022

MIDTERM 1

Student Name: Ernesto Ibarra
Student #: 2001211571
Student Email: ibarre3@unlv.nevada.edu
Primary Github address: https://github.com/ErnestoIbarra333
Directory: https://github.com/ErnestoIbarra333/ErnestoIbarra

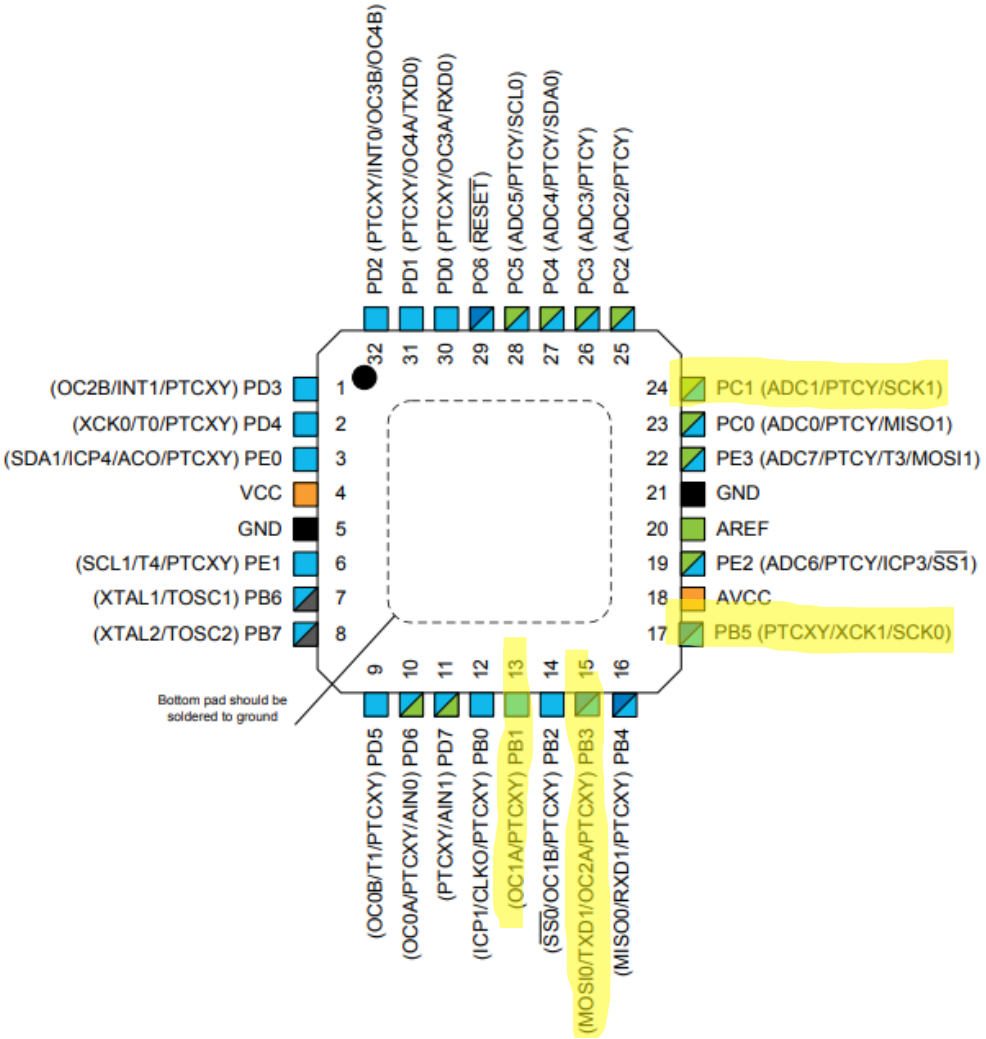
1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

Atmel Studio 7.0
- Assembler
- Simulator
- Debugger

Atmega328PB-Xmini

Multi-Function Shield
- Switches
- LEDs

Logic Analyzer



2. INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1-6

```
/*
 * MIDTERM1.c
 *
 * Created: 3/20/2022 11:00:18 am
 * Author : Ernesto Ibarra
 */

#define BAUD 9600
#define F_CPU 16000000UL
#include <avr/io.h>
#include <stdio.h>
#include <stdbool.h>
#include <stdio.h>
#include <stdlib.h>
#include <util/delay.h>
#include <avr/interrupt.h>

void initTimer1CTC();
void startTimer1CTC_OC();
void stopTimer1CTC_OC();
void USART_init(void);
void USART_send(volatile char *data);
char USART_Receive();
void USART_putstring(char* StringPtr);

volatile char *help =
"1. ‘o’ - turns ON LED at PB5.           ‘O’ turns OFF the LED at PB5.  \n"
"2. ‘p’ - Blink (on-off) the LED PB3.    ‘P’ turns off the LED PB3. \n"
"3. ‘f’ - fade the intensity of LED PB1. ‘F’ turns off the LED PB1/stops this operation.
\n"
"4. ‘b’ – reads the status of the switch at PC1. Display the status in terminal. Exit
this function when there is change of switch status. \n""\n";

int main(void)
{
       cli();
       DDRB |= (1<<1); // set PORTB1 for output
       DDRB |= (1<<3);      // set PORTB3 for output
       DDRB |= (1<<5); // set PORTB5 for output
       PORTB |= (1<<3);// set LED off to start
       PORTB |= (1<<5);// set LED off to start
       PORTB |= (1<<1);// set LED off to start
       DDRC &= (0<<2); // here we set portC2 as in input
       PORTC |= (1<<2); //enable pull up resistor
       USART_init(); // Initializes the ADC
       USART_send(help); // here we print out all the options at the start
       sei(); // enable interrupts

    while (1)
    {

    }
}
```

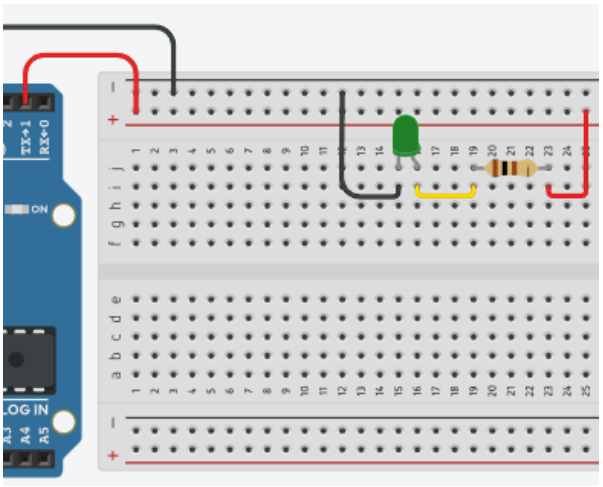
```
ISR(USART0_RX_vect)
{
	char data = USART_Receive();
	if (data == 'o')
	{
		bool check = true;
		while(check)
		{
			PORTB &= ~(1<<5); // LED ON
			data = USART_Receive(); // here we check for new input
			if(data == 'O') // if big O we will turn off LED
			{
				check = false;
			}
		}
	}
 else if (data == 'f')
 {
	 bool check3 = true;

	 DDRB |= (1 << 2); //pb2 is our output since pb1 has no onboard lED
	 TCCR1A |= (1 << COM1A1) | (1 << COM1B1); // set our settings for fast PDW
	 TCCR1A |= (1 << WGM11);
	 TCCR1B |= (1 << WGM12) | (1 << WGM13);
	 TCCR1B |= (1 << CS10); // we start off with no prescalar
	 ICR1 = 0xFFFF; // our top is 0xFFFF
	 OCR1A = 0xBFFF; // this is the duty cycle of 75%
	 OCR1B = 0x2FFF; //duty cycle of 19%
	 int counter1 = 0;
	 while (check3 == true)
	 {
		 data = USART_Receive(); // here we check for another input f
		 if (data == 'f')
		 {

			 if (counter1 == 0)
			 {
				 OCR1B = 0x5FFF; // duty cycle decreased
				 counter1 = counter1 + 1; // here we increment counter
			 }
			 else if (counter1 == 1)
			 {
				 OCR1B = 0xBFFF; // duty cycle decreased again
				 counter1 = counter1 + 1; // here we increment counter
			 }
			 else
			 {
				 counter1 = 0;
			 }
		 }
		 if (data == 'F') { // given bi F we will end the fading
			 OCR1B = 0xFFFF; // here we turn the LED
			 check3 = false;
		 }
	 }
 }
	else if( data =='p')
```

```
	{
		TCCR1B |= (1<<WGM12) | (1<<CS12); // here we use a timer to blink LED3
		TCNT1 = 0; // start it from zero
		OCR1A = 4000; // our top value will be 4000
		TIMSK1 |= (1<<OCIE1A); // enable match interrupt
	}
	else if(data =='P')
	{
		TCCR1B &= (0<<CS12) | (0<<CS11) | (0 << CS00); // here we turn off the
timer
		PORTB |= (1<<3);	// turn off the LED
	}
	else if( data == 'b')
	{
		int mode = 0;
		bool check = true;
		if(!(PINC & (1<<PINC2))) // here we start off with either LED on or off
		{
			USART_send("Switch is High \n");
			mode = 2;
		}
		else
		{
			USART_send("Switch is Low \n");
			mode = 1;
		}
		_delay_ms(1000);

		if(mode == 1) // here it means the LED started off off
		{
			while(check) // we keep looping until switch is toggled
			{
				if(!(PINC & (1<<PINC2)))
				{
					USART_send("Switch is High \n");
					mode = 2;
				}
				else
				{
					USART_send("Switch is Low \n");
					mode = 1;
				}
				if(mode == 2) // once the switch changes status we will exit
loop.
				{
					check = false;
				}
				_delay_ms(1000);
			}

		}
		if (mode == 2)
		{
			while(check)// we keep looping until switch is toggled
			{
				if(!(PINC & (1<<PINC2)))
				{
					USART_send("Switch is High \n");
```

```
                                mode = 2;
                        }
                        else
                        {
                                USART_send("Switch is Low \n");
                                mode = 1;
                        }
                        if(mode == 1)// once the switch changes status we will exit
loop.
                        {
                                check = false;
                        }
                        _delay_ms(1000);
                }
        }

    }
    else if( data == 'h')
    {

        USART_send(help);
    }

}

ISR(TIMER1_COMPA_vect) // interrupt used for blinking LED3
{
    PORTB ^= (1<<3);
}

void USART_init( void ) // initialize the USART with required settings
{
    UBRR0H = 0; // high byte of UBRR0
    UBRR0L = (F_CPU/16)/BAUD - 1; // low byte of UBRR0
    UCSR0C = _BV(UCSZ01) | _BV(UCSZ00); // this is for 8 bits
    UCSR0B = _BV(RXEN0) | _BV(TXEN0) | (1<<RXCIE0); // here we enable RX and TX
    TCCR1B |= 0; // Sets pre scaler to 1024////////////////////////////////////
    TIMSK1 = (1 << TOIE1); // Enable overflow flag
    TCNT1 = 49911; // 1 second delay = 49911
}

void USART_send(volatile char * data) // function to send to USART
{
    while((*data!= '\0'))
    {
        while (!(UCSR0A & (1 << UDRE0))); // loops until UDRE0 is set
        UDR0 = *data; // UDR0 register stores the value pulled from data
        data++;
    }

}

 char USART_Receive( void ) // function to receive using USART
{
    while ( !(UCSR0A & (1<<RXC0)) ) // loops until all data is read.
    {
    }
    return UDR0; // returns UDR0
```

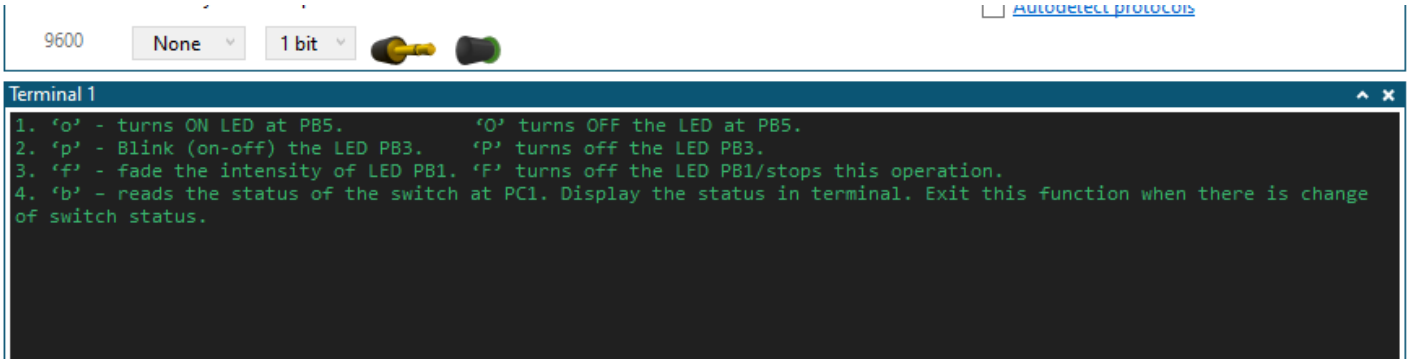
```
}

void USART_putstring(char *StringPtr) // puts data into a string
{
	while ((*StringPtr != '\0')) // loops until line is empty
	{
		while (!(UCSR0A & (1 << UDRE0))); // loops until UDRE0 is set
		UDR0 = *StringPtr; // UDR0 will store the value of string ptr
		StringPtr++; // will increment until entire string is done
	}
}

void initTimer1CTC() //Here we set the pre scalar and the delay we want in hertz
{
	TCCR1B |= (1U<<3);
}

void startTimer1CTC_OC() // here we start our timer with the value or zero in CTC
{
	TCNT1 = 0x00;
	TCCR1B |= (0x05<<0);
}

void stopTimer1CTC_OC() // here we are stopping the timer.
{
	TCCR1B &= ~(0x07<<0);
}
```

3. SCHEMATICS

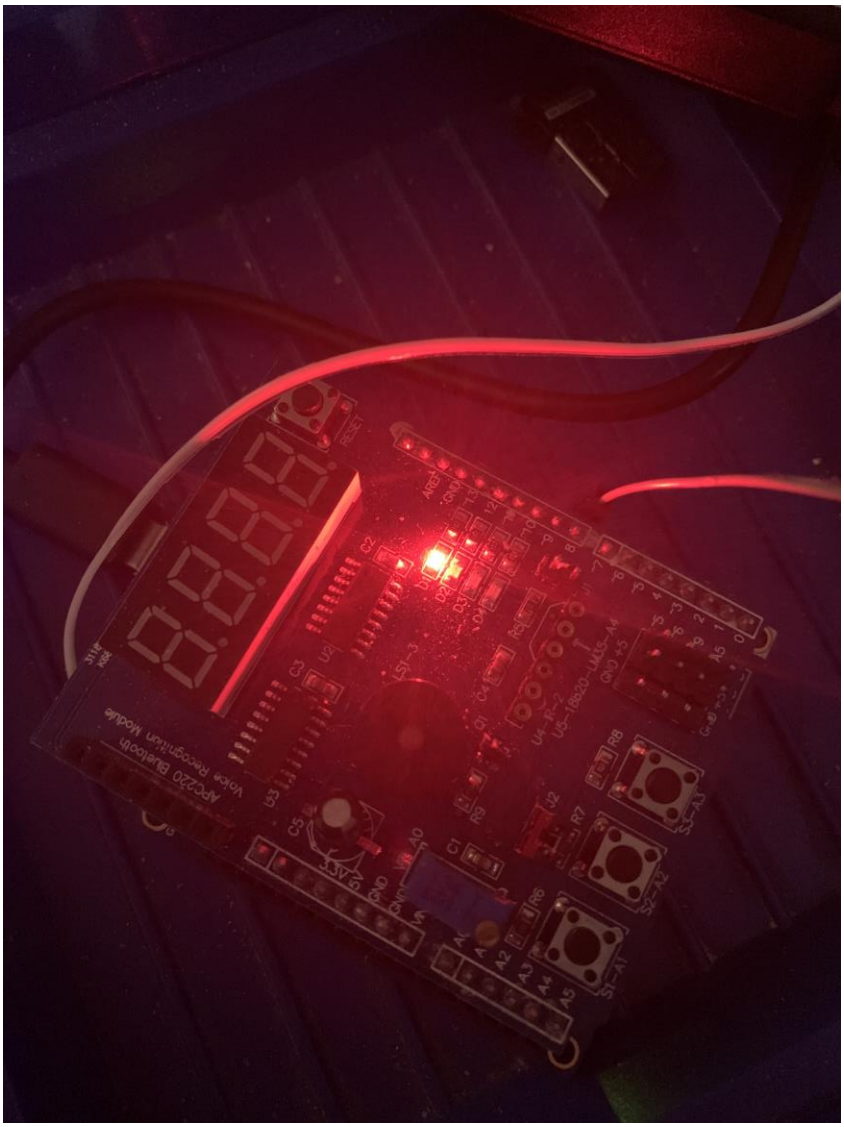


4. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

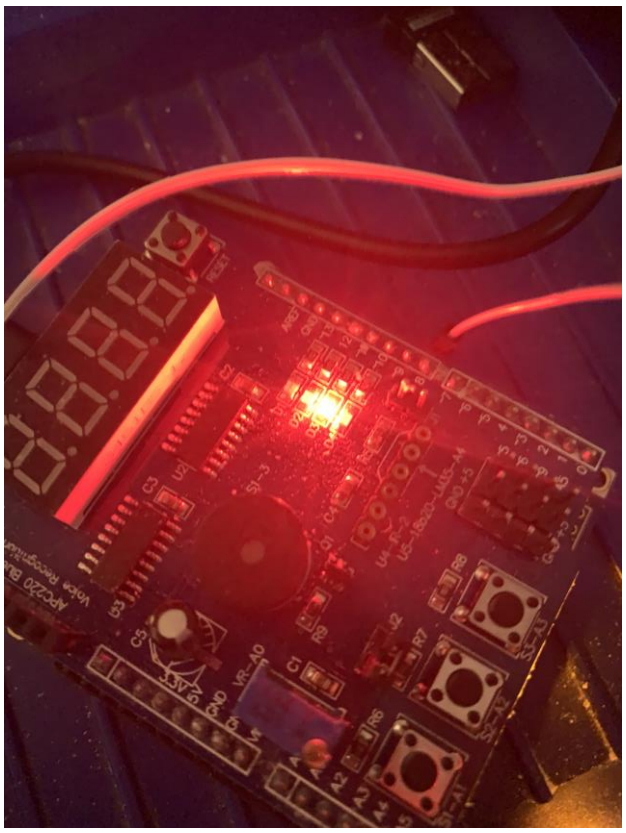
Here we start off with task H which means it will display all the different inputs the user can give to the board.



Now we move on to 'o' and 'O'. This will turn on LED at PB% and leave it on until you press 'O' then the LED will turn off.

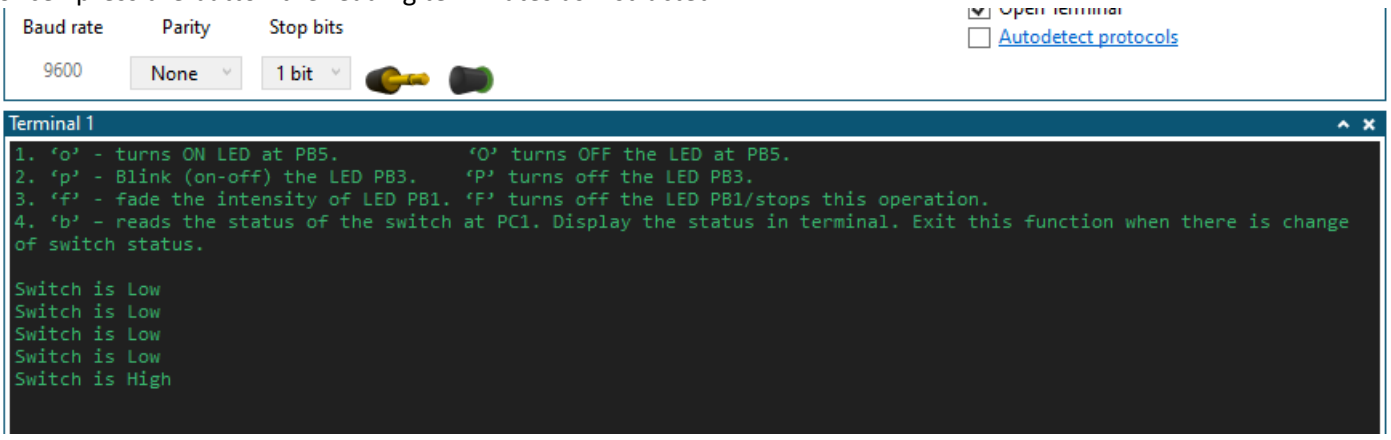


Now we will move on to p which will make the LED at PB3 turn on and off constantly until you press ‘P’ to turn off the LED all together.

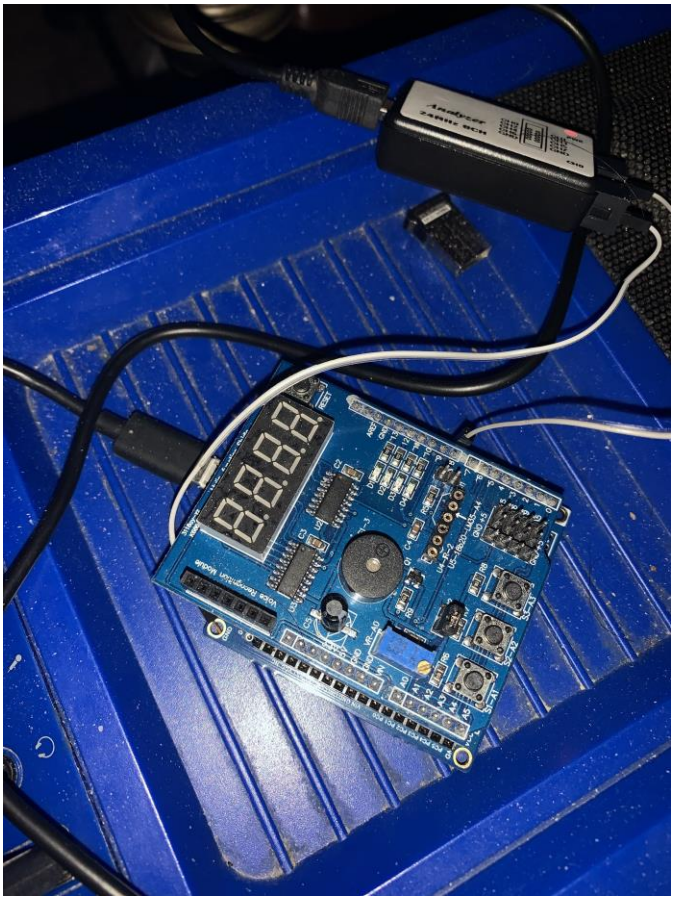


For this step I will show an LED fading on the board, I will skip the picture and will cover this step on the video.

Finally, we have the ‘b’ and ‘B’ which will cause the board to read the status of the switch at PC1 and display the current state on the board. It will terminate once a new change of switch status has been detected. Below we can see the switch starts in the OFF state which is why it keeps reading it and then once I press the button the reading terminates as instructed.



5. SCREENSHOT OF EACH DEMO (BOARD SETUP)



6. VIDEO LINKS OF EACH DEMO

https://youtu.be/OGbhelBXVCk

7. GITHUB LINK OF THIS DA

Student Academic Misconduct Policy
http://studentconduct.unlv.edu/misconduct/policy.html

"*This assignment submission is my own, original work*".
Ernesto Ibarra-Ayala